1. ВВЕДЕНИЕ

2. ТЕОРЕТИЧЕСКОЕ ОПИСАНИЕ МОДЕЛИ

Рассмотрим модель

$$dS_t = f\left(M_t\right)dt + \sigma_t dW_t \quad \overset{def}{\Leftrightarrow} \quad S_t = \int\limits_0^t f\big(M_s\big)ds + \int\limits_0^t \sigma_s dW_s, \quad (1)$$

где S_t - логарифм зависимой переменной, f - некоторая функция от M_t - матрицы "объекты-признаки", σ_t - параметр, в общем случае непостоянный, W_t - броуновское движение.

Заметим, что выражение (1) можно представить в виде

$$S_{k\Delta} - S_{(k-1)\Delta} = \big(f\big(M_{k\Delta}\big) - f\big(M_{(k-1)\Delta}\big)\big)\,\Delta + \sigma_{k\Delta}\left(W_{k\Delta} - W_{(k-1)\Delta}\right), \quad (2)$$

где $k\Delta$ и $(k-1)\Delta$ - некоторые моменты времени на сетке $\{\Delta, 2\Delta, ..., n\Delta\}$. Тогда, деление обеих частей (2) на $\sqrt{\Delta}$ даст

$$\frac{S_{k\Delta} - S_{(k-1)\Delta}}{\sqrt{\Delta}} = \big(f\big(M_{k\Delta}\big) - f\big(M_{(k-1)\Delta}\big)\big)\sqrt{\Delta} + \sigma_{k\Delta}\frac{\big(W_{k\Delta} - W_{(k-1)\Delta}\big)}{\sqrt{\Delta}}. \quad (3)$$

Также известно, что $W_{k\Delta} - W_{(k-1)\Delta} \sim \mathrm{N}(0,\Delta)$. Следовательно, $\frac{W_{k\Delta} - W_{(k-1)\Delta}}{\sqrt{\Delta}} \sim \mathrm{N}(0,1)$.

Поэтому

$$\frac{S_{k\Delta} - S_{(k-1)\Delta}}{\sqrt{\Delta}} = \big(f\big(M_{k\Delta}\big) - f\big(M_{(k-1)\Delta}\big)\big)\sqrt{\Delta} + \sigma_{k\Delta}Z_{k\Delta}, \quad Z_{k\Delta} \sim \mathcal{N}(0,1). \quad (4)$$

3. ОЦЕНКА ФУНКЦИИ f

3.1. Рекуррентные нейронные сети

Для обработки последовательных данных зачастую используют рекуррентные нейронные сети (далее RNN - recurrent neural networks). Их ключевой особенностью является использование при обучении предшествующих данных наряду с текущими, что схематично представлено на Рис. 1.

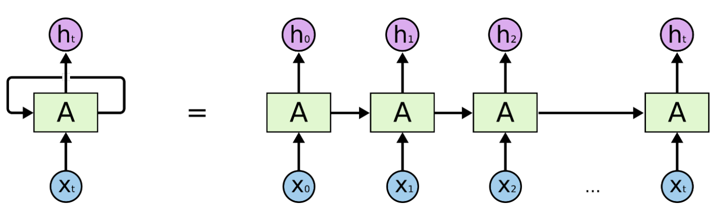

Рис. 1: Структура простейшей RNN

В общих чертах принцип работы простейшей RNN подразумевает сохранение результатов обучения на предыдущих относительно текущего значениях и их использования для более качественного прогнозирования. Действительно, в контексте временных редов и последовательных данных в целом история очевидно играет значительную роль, поэтому такой подход является вполне разумным.

При этом базовая структура RNN может повлечь несколько значительных проблем при обучении, например, таких как **взрыв** или **затухание градиентов** (vanishing or exploding gradients). Проще всего ее осознать на конкретном примере. Так, на Рис. 2 представлен частный случай RNN, в котором за передачу информации от предыдущего значения к текущему отвечает параметр W_2. Также понятно, что при обратном проходе по графу (поиске производных по параметрам) в цепочке произведения частных производных мы так или иначе получим W_2^2. Следовательно, при числе слоев n мы столкнемся с множителем W_2^n, и тогда если $|W_2| < 1$, то $W_2^n \to 0$ при $n \to \infty$, что называется затуханием градиента. Аналогично при $|W_2| > 1$ $W_2^n \to \infty$ при $n \to \infty$ - взрыв градиента.

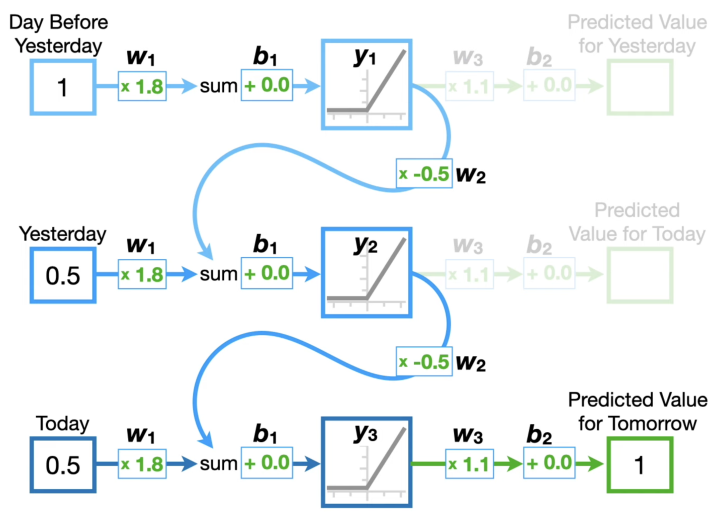

Рис. 2: Частный пример RNN [1]

Заметим, что выявленные проблемы действительно значительны, так как при затухании градиента веса фактически перестают обновляться, а при взрыве - наоборот меняются слишком сильно и препятствуют нахождению оптимума, что следует из формулы

$$w_t = w_{t-1} - \frac{1}{\eta} Q(w_{t-1}), \quad (5)$$

где w_t - вектор весов в момент t, η - скорость обучения, $Q(x)$ - градиент функции потерь.

Таким образом, для решения нашей задачи необходимо более продуманная реализация RNN, не имеющая вышеописанных недостатков, например, LSTM (Long Short-Term Memory), которую мы рассмотрим в дальнейшем.

3.2. Long Short-Term Memory

Перейдем к описанию следующего поколения RNN - модели LSTM (Long Short-Term Memory). Данная модификация простейшей рекуррентной нейронной сети позволяет избежать такой значительной проблемы, как затухание или взрыв градиента засчет специальной структуры, схематично представленной на Рис. 3.

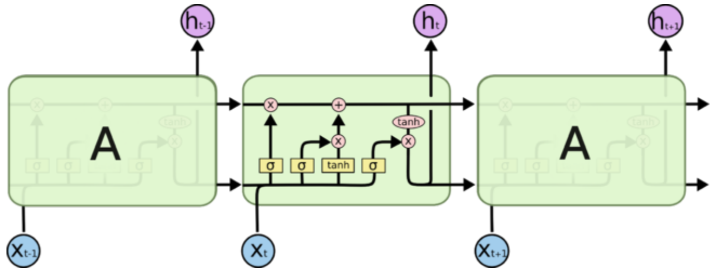

Рис. 3: Структура LSTM

Основная идея LSTM-сетей заключается в использовании двух потоков информации, влияющих на конечный прогноз напрямую - краткосрочной и долгосрочной памяти (Рис. 4). Заметим, что в базовой RNN долгосрочные данные имели лишь опосредованное значение через последовательное суммирование и применение функций активации (Рис. 2).

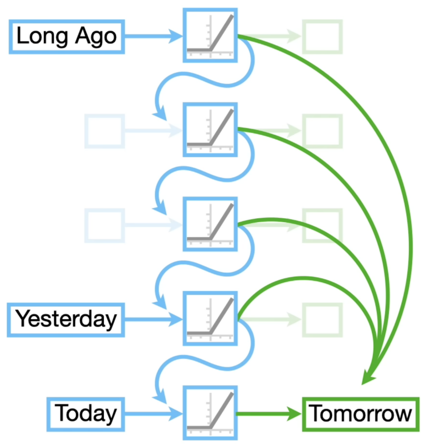

Рис. 4: Иллюстрация влияния долгосрочной и краткосрочной памяти в рамках LSTM-модели[2]

Наконец перейдем к более подробному описанию LSTM. Для начала еще раз обратимся к Рис. 3, на котором видно, что в рамках данной модели используются две функции активации - сигмоида (σ) и гиперболический тангенс ($tanh$). Вспомним, как они задаются

1. Сигмоида:
$$\sigma(x) = \frac{e^x}{1+e^x}, \quad (6)$$
причем $\sigma : \mathbb{R} \mapsto (0, 1)$.

2. Гиперболический тангенс:
$$tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}}, \quad (7)$$
причем $tanh : \mathbb{R} \mapsto (-1, 1)$.

Что касается одной составляющей LSTM-модели, то ее условно можно разделить на три блока, представленных на Рис. 5, в котором розовой линией обозначена краткосрочная память, а зеленой - долгосрочная. Также функция в маленьком голубом прямоугольнике есть сигмоида, а в маленьком оранжевом - гиперболический тангенс.

Первый блок выделен большим голубым прямоугольником, в котором по существу происходит определение доли от входящего значения *долгосрочной* памяти, которую необходимо оставить для дальнейшего использования.

Второй блок образован большим зеленым и большим оранжевым прямоугольниками. В большом оранжевом прямоугольнике вычисляется потенциальная *долгосрочная* память, а в синем - аналогично первому блоку доля, которую модель запомнит. Затем найденное значение будет сложено с получившимся после первого блока, что образует новую *долгосрочную* память, которая будем передана в дальнейшие итерации модели.

Третий блок получается из объединения большого фиолетого и большого розового прямоугольников. В нем и образуется выход модели, который либо передается в последующие части модели, аналогичные текущей, в качестве новой *краткосрочной* памяти, либо уже представляет конечный результат. Более конкретно, в третьем блоке на основе новой *долгосрочной* памяти определяется потенциальная *краткосрочная*, и по известной схеме из первого блока определяется ее доля.

Заметим, что образование *долгосрочной* памяти не имеет весов, что как раз и является решением проблемы затухающих или взрывающихся градиентов.

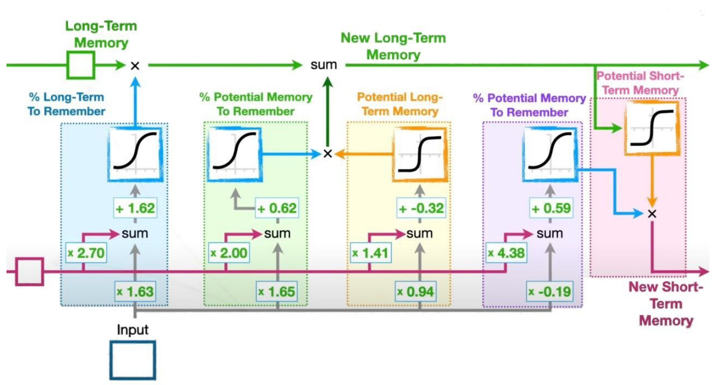

Рис. 5: Частный пример LSTM[2]

Таким образом,

NB: Сказать, что зеленая линия не имеет везов, следовательно, нет проблемы с градиентами. Про то, что на выход можно навесить полносвязный слой

4. Источники

[1] StatQuest with Josh Starmer: Recurrent Neural Networks (RNNs)

[2] StatQuest with Josh Starmer: Long Short-Term Memory (LSTM)